アルゴリズム 35－5
（1993．10．1）

BDDを用いたマンハッタン配線問題の解法

梅田達也　戸川望　佐藤政生　大附辰夫

早稲田大学理工学部
〒169 東京都新宿区大久保 3-4-1

マンハッタン配線問題において、交差なしに結線可能な最大ネット集合を求める問題はNP困難な問題である。本稿では、この問題をBDDを用いて解く手法を示す。さらに、この手法では、BDDの変数順が重要になるため、本稿の問題に適した変数順の決定法に関して考察する。また本手法に適した問題の性質についても考察する。

A Method for Manhattan Wiring using BDD

Tatsuya Umeda, Nozomu Togawa , Masao Sato, and Tatsuo Ohtsuki

School of Science &Engineering, Waseda University
3-4-1 Okubo, Shinjuku, Tokyo 169, Japan

To find the maximum number of nets which can be connected by manhattan wires without cossing on a place is NP-hard. This problem is considered as a prblem of finding MIS(Maximum Independence Set) in graph theory. This paper presents a method for finding MIS using BDD(binary decision diagram). The variable ordering of BDD is quite important.Thus we consider suitable variable orderings for Manhattan wiring. In this paper we also concern about the condition in which our method could be effective.

1　はじめに

論理関数を表現する手法として，BDD（二分決定グラフ:Binary Decision Diagram）というグラフ表現が用いられることが多くなってきている[1]．BDDは他の論理表現手法と比べて記憶効率が良いため，従来の論理表現よりも大規模な論理関数を操作する事が可能になる．

組み合わせ最適化問題を解く方法の一つとして，BDDによる論理表現を用いる手法がある．この方法は問題の制約式をBDD表現し，そのグラフ上での最短路探索によって最適解を得るという方法である．BDD上での最短路探索に要する手間は，対象となるその節点数に比例することより，問題の制約を現実的節点数のBDD表現を現実的時間内に得る事が可能であれば，その問題はこの手法によって現実的時間で解ける事になる．冒頭で述べたようにBDDは記憶効率に優れるため，従来扱えなかった規模な組み合わせ最適化問題でも，制約をBDDによって現実的サイズで表現できる可能性がある．従って，この手法によれば，従来手法では時間的や空間的な理由で解けなかったような難しい問題を，多項式時間で問題が解ける可能性がでてくる[2]．

結線要求のある二端子ネットの集合を高々１曲がりの配線によって平面上で交差することなしに結線する問題をマンハッタンワイヤリング問題という．このマンハッタンワイヤリングに関する問題のうち，交差なしで結線できる最大のネット集合を求める問題はNP困難な問題であることが知られている[3]．本稿では，この問題を上記のグラフを用いた手法に基づいて解く方法を提案する．

2　ＢＤＤ（二分決定グラフ）

BDDはサイクルのない有向グラフによる論理関数の表現手法である．図1に論理式とそれに対応するBDDを示す．四角い節点を定数節点といい，その中の値は関数の論理値を表す．丸い節点は変数節点といい，その中の数字は論理変数番号を表す．また一番上の節点を根節点と呼ぶ．枝に付記した数値は節点内の論理変数の論理値である．

BDDには二つの大きな利点がある．まず一つ目は，BDDは論理的に冗長な節点は省略し，論理的に等価な節点は一つにまとめるため，記憶効率がよい．このため従来扱えなかった規模の論理関数を操作する事が可能になる．二つめは親節点とその子節点の変数番号の大小関係を固定する制約により，論理関数に一意に対応する表現となる．

またBDDの構築は，Bryantのapply演算によって，論理式から効率的に実行する事が可能である[1]．

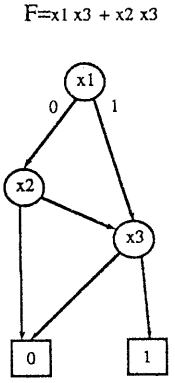

図1　BDD

3　マンハッタン配線問題

マンハッタン配線とは，図2のような結線要求のある二端子ネットの集合があったとき，これらのネットを高々一曲がりの配線によって平面上で交差することなしに結線することをいう．このマンハッタン配線に関する問題のうち，全ネットを交差なしで結線することが可能かどうかを判定する問題に対しては，O(n log n)のアルゴリズムが提案

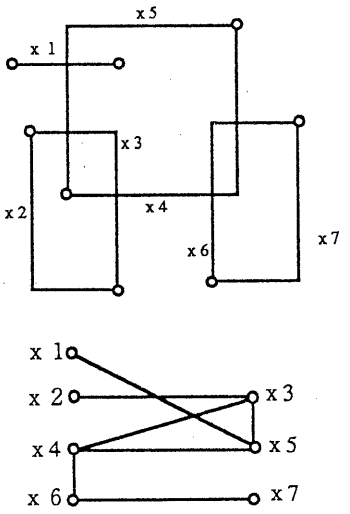

図2　ManhattanWiring

されている[3][4]．但しnはネット数である．しかし，完全な結線が不可能と分かった時に，交差なしに結線できる最大のネット集合（ネット数）を求める問題は，NP困難な問題である[5]．この問題は，各配線を節点，配線間の交差を枝に対応させたグラフにおいてMIS（Maximum Independence Set:最大節点独立集合）を求める問題に等しい．本稿では，このMISを求める問題をBDDに基づいて解く方法について考える．

4　BDDを用いた解法について

4.1　BDDによる組み合わせ最適化問題の解法の一般的な形

組み合わせ最適化問題のうち，NP完全やNP困難に属する問題は，従来の解法（例えば分枝限定法など）では問題のサイズが大きくなってくると，現実的時間内に最適解を求めるということは困難になってくる．このような大規模な問題に対してBDDに基づいた手法を適用することによって，それらのうちの幾つかを現実的時間内に解く事が可能となる．　このBDDによる方法の一般的な形を以下に示す[2]．

（1）問題の定式化
まず問題の制約式を論理式によって表現する．組み合わせる要素iの選択／非選択を論理変数xiで表現し、論理ベクトル(x1，x2，・・，xi，・・，xn)によっあらゆる組み合わせを表現する．このとき組み合わせが満たさねばならない制約を

$$F(x1,x2,\cdot\cdot,xi,\cdots xn) = 1$$

という形の論理式で表現する．次に，要素iのコストをwiとし，最小化（あるいは最大化）するコスト関数を次のような形に表す．

$$C = \Sigma xi \cdot wi$$

つまり，求める解は，式Fを満足する(x1，x2，・・，xi，・・，xn)のうち，Cを最小化する(あるいは最大化する）ものである．
（2）BDDの構築
制約を表す論理式のBDDを構築する．
（3）最適解の探索

BDDの各枝に，重みxi･wiを対応させ，グラフの根から定数節点1へ至る最短路（あるいは最長路）を求める事によって，コストを最適化するxiへの割当を求める．（このとき最短路が変数xiの節点から1-枝を経るならその変数の要素は選択、0-枝なら非選択とする．ただし節点が省略されている要素は全て選択と考える．）

4.2　マンハッタン配線問題へのBDD法の適用

ここでは，マンハッタン配線問題に，前節で示したBDDによる方法をあてはめる．

4.2.1　問題の定式化

本稿で扱う問題は交差無しで結線できる最大のネット数を求めるというものである．まず各配線に論理変数を割り当て，配線の選択／非選択を論理値1/0によって表現し，論理ベクトル(xi)によってあらゆる結線方法を表現する．求める最適解は，より多くの1を含む論理ベクトル(x1，x2，・・，xi，・・，xn)であるから，コスト関数は，

$$C = \Sigma xi$$

となり，目的はこの関数の最大化になる．制約条件は，(x1，x2，・・，xi，・・，xn)による組み合わせが互いに交差する配線を同時に結線してはならないことである．この制約条件を表す論理式は，互いに交差する配線の論理変数のNANDをとり，さらにあらゆる交差ペアに対するこのNANDの論理積を取る事によって得られる．例えば配線aとbおよび，cとdが交差するネットであれば，以下の論理式

$$F = \neg (a \cdot b) \cdot \neg (c \cdot d)$$

によって制約式が表現される．また図2のネット例に対する制約式は次の論理式によって表現される．

$$\begin{aligned} F = {} & \neg (x1 \cdot x5) \cdot \neg (x4 \cdot x6) \\ & \cdot \neg (x3 \cdot x2) \cdot \neg (5 \cdot x4) \\ & \cdot \neg (x4 \cdot x3) \cdot \neg (x3 \cdot x5) \cdot \neg (x6 \cdot x7) \end{aligned}$$

$$= \neg \{ x1 \cdot x5 + x2 \cdot x3 +$$

x3·x4 +x3·x5 + x4·x5 + x4·x6 + x6·x7 }

= 1

4.2.2 BDDの構築

4.2.1で生成した制約条件を表す論理式から，その論理を表すBDDを構築する．一般的にBDDには，論理式に頻繁に現れる変数はBDDの上位に位置付けるとBDD節点数が減る，互いに近い位置にある変数はBDD上でも近い位置に置くと節点数が減る，という性質があるので，これらを考慮してBDDを構築する．BDDの構築においてはBEM（Boolean Expression Manipulator）[6]をはじめ多くのBDDツールが記憶効率の向上や，BDD構築時間の短縮のために否定枝，変数シフト枝，入力反転枝などの属性枝を用いている[6]，[7]．しかし本稿で扱う問題は制約式が非常に単純な事もあり，否定枝をはじめ変数シフト枝，入力反転枝などを用いなくても，記憶効率などが損なわれることはほとんどないと考えられるのでこれらの属性枝は用いなかった．

図2のネット例に関しては次のようなBDDが構築できる．

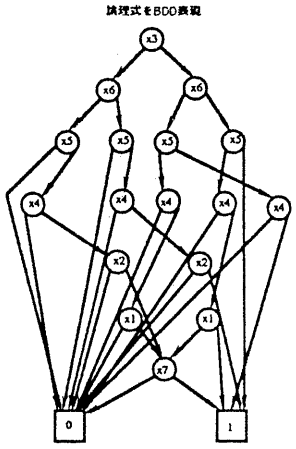

図3　図2のネットに対するBDD

4.2.3 最短路探索

MISを実現する論理ベクトル(x1，x2，・・，xi，・・，xn)は，4.2.2で示した論理式を満たすもののうち，1に割当てられた変数がもっとも多いものである．この割当ては，BDD上で1-枝に重み0，0-枝に重み1を与えたときの，定数節点1から根節点への最短路として与えられる．すなわち，最短路上で1-枝を経る変数に対応する配線　と枝が飛び越している変数の配線のみを結線すると，最大ネット集合を得ることができる．

最短路探索の手続きを下に示す．下の手続きを根節点に適用すると，各節点の1節点からの最短距離が深さ優先に計算される．vはBDDの各節点，high(v)，low(v)はそれぞれvの1-枝，0-枝の差す節点である．v.sonは最短路の行き先を示す．v.sonを辿ることによって最短経路が得られる．この手続きは明らかに，BDDの節点数に比例する手間で実行できる．

径路探索の一例を図4に示す．

```
weight(v){
        if(v≠定数節点){
            highweight =weight(high(v));
            lowweight =weight(low(v));
            if ( highweight < lowweight+1] )
                v.son = high(v);
            else
                v.son  =low(v);
            return min[hjghweight,lowweight+1];
        } else if(v == 0節点)
            return ∞;
        else if(v == 1節点)
            return 0;
}
```

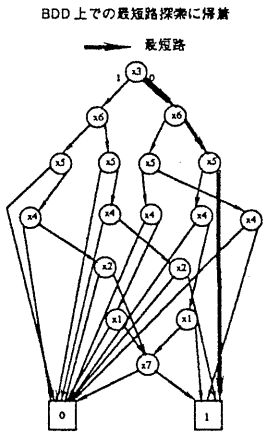

図4　BDD上での最短路探索

5 BDDによる手法の優位性

本稿で述べた手法の明確な利点としては、とける問題の規模が増大したことのほかに，この手法におけるBDDがMISのみでなく，全ての可能な結線組み合わせをグラフ内に保持しているという点が挙げられる．これによって，様々な要求に対応して，最も適当な解をBDDから選択することが可能になると考えられる．

6 実験結果

本章では，BDDを用いた手法によって実際にマンハッタン配線問題を解いた結果を示す．計算機はSparc Station 2 (28.5MIPS)を用い，BDDはBEM(ただし，属性枝は用いなかった)によって構築した．BDD構築においては，いくつかの効率化の方法が考えられるので，それらをまじえて，結果を示す．

6.1 共通因子による効率化

制約を表す論理式は，例えば下のように一項目の共通因子をくくりだすことができる．

$$F = \neg \{ x1 \cdot x5 + x2 \cdot x3 + x3 \cdot x4 + x3 \cdot x5 + x4 \cdot x5 + x4 \cdot x6 + x6 \cdot x7 \}$$

$$= \neg \{ x1 \cdot x5 + x2 \cdot x3 + x3 \cdot (x4 + x5) + x4 \cdot (x5 + x6) + x6 \cdot x7 \}$$

このようにして共通因子をくくりだすと，BDD構築の際に要する二項演算の回数が減るため，処理時間を1/2から1/3程度に減らすことができる．

表1:共通因子による効率化

BDD節点	処理時間(sec)	
	共通因子の考慮なし	共通因子を考慮
99540	113.07	53.77
126841	92.30	52.24
42232	50.49	56.51
61078	83.92	36.31
43478	54.89	27.72

その効果を表1に示す．また実験対象には乱数によって端子を生成した五つのネットを例題として用いた．またネット数は40である。

6.2 変数順序

6.2.1 変数順序とBDDの節点数

BDDでは先に述べたように，親節点と子節点の変数番号の大小関係を固定する，すなわち，論理変数にある順序を与える．そしてBDDの節点数はこの変数順序によって変動する．従って，BDDを構築する際に，そのサイズを小さくして効率の良い論理表現とするためには，できるだけ良い変数順序を見つけなくてはならない．（しかし最良の変数順序を発見するには，指数時間の手間がかかるため，関数の性質などを考慮したヒューリスティック手法を用いるのが一般的である[8][9][10]．）

一方，BDDを用いて組み合わせ最適化問題を解く場合，その処理時間のほとんどがBDD構築時間によって占められるため，記憶効率の点からはもちろん，処理時間の点からもBDDのサイズを小さく抑える事は特に重要である．

そこで以下では，本紙で扱う問題に適したヒューリスティックな変数順序付けについて述べる．

6.2.2 交差数に基づく順序づけ

一般に論理式に頻繁に現れる変数は関数の論理値を制御する力が強いので，これらに対応する節点からは定数節点へつながる枝が多い．このためこれらの変数を，BDDの上位に位置づけると，BDDの構造は簡単になり，節点数は低くなる傾向を持つ．従って，本問題においては，交差数の多い配線を表す変数をBDDのより上位に位置づけることで，節点数を減らすことができる．表2にこの順序付け法を，前節で用いた物と同じ5つの例題に対して適用した結果を示す．この結果を見ると、特に順序付けを考慮しなかった場合（この場合，論理式中に先に現れた変数から順に上位に固定される）と比べて節点数が1/2程度に減っている事が分かる．

6.2.3 交差関係に基づく順序づけ

論理式上で互いに近い位置にある変数同士（例えば互いに一つの積項をなす変数同士）はそれら

一組で関数値を制御する力をもつ．従ってこうした変数同士は，BDD上で遠い順序に置くよりも，近い順序に置いた方が，それに対応する節点から定数節点へ直接つながる枝が多くなり，BDDの構造は簡単になることが知られている．このため，本稿で扱う問題においては，互いに交差している配線の変数を近い位置に置くことによってBDDの節点数を減らすことができる．表3に交差関係による順序付けの結果を示す．対象としたネットは前節と同じである．交差関係による順序付けは，上位から順に固定する方法を取った．最上位に来る変数は，交差数の最も多いものに決定し，それよりも下位の変数はより上位に固定された変数との交差数の多いものから固定した．

結果からこの順序付けによって節点数を1/2から1/4に程度減らせることが分かる．

6.2.4 交差数と交差関係を同時に考慮した順序づけ

6.2.2と6.2.3に示した順序づけの規準は，本質的に異なる規準である．従ってこれらの基準を完全に両立させて考慮する事は難しい．しかしこれらに適当な優先度を与えて，二つの規準を同時に考慮すれば，よりよい変数順序を発見できる可能性がある．そこで，これらの規準に様々な重みを与えて，順序を生成し，その順序のもとでBDDを構築した．具体的には，各変数に対し，次式による評価値を計算し，それによって順序を決めた．

$$w1 \times (CROSS/CROSSMAX) + w2 \times (UCROSS/UCROSSMAX)$$

ただし，

CROSS = 交差数
CROSSMAX = 最大交差数
UCROSS = 上位に対する交差数
UCROSSMAX = 上位に対する交差数の最大値

である．
結果は表4のようになった．
（これまでのように，乱数によるネットを用いると，問題例によって結果が変動し，基本的な特徴をつかみにくいので、図5のような規則的な構造の40ネットを例として扱った．）
この結果より，交差数による順序づけの規準よりも，互いに交差するものを近く置く規準の方が，BDD節点数を減らす効果が高いと考えられる．

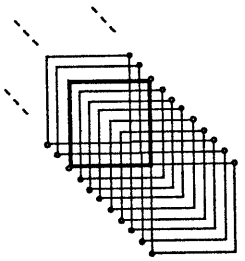

図5 規則的構造のネット

6.2.5 配線の位置に基づく順序付け

前節の結果を見ると，交差数に基づく規準はこの問題に関してあまり効果がなく，互いの交差関係の規準のみを考慮した場合がもっとも節点数が少ないことが分かる．従って本問題に関しては交差関係のみを考慮した順序付けが適当であるといえる．ところが互いに交差するものを近く置く基準のみを考慮すると，結局は位置的に近い配線を表す変数がBDD上でも近い順位になると考えられる．そこで，各配線間の交差関係を考慮するのではなく，単純に配線の位置を表す座標の大きい順に順序づけを行うことで，互いに交差するものを近い順位におく方法が考えられる．この方法による順序づけを用いると，BDDの節点数は表5のような結果となった．順序付け考慮なしの場合の結果も同時に示す．（実験は，前節で用いた5つの例題を対象として行った）この結果を見ると，配線の位置に基づく順序づけが非常に効果的であることが分かる．これまで述べてきた順序づけ法の中でも，この方法がもっとも効果的といえる．

6.3 ネットの交差数とBDD節点数の関係

図6のような定数交差ネットによって様々な交差数のネットを生成し，その交差数とBDDの節点数との関係を調べると以下の表6のような結果となった．

この結果をみると，比較的交差の多い密集したネットに対して，BDDの節点数が少なくなる傾向にある事が分かる．このことは，問題の解の個数と関係があると考えられる．制約関数のBDDは，MISに限らず交差を含まないあらゆる解を保持している．一方，解の個数は密集したネットほど少なく，まばらなネットになると解の個数が多くな

る．従って，ネットがまばらになるほど，それに対するBDDは多くの解を保持しなくてはならなくなり，結果的にBDDの節点数が多くなると考えられる．よって，本紙で提案したBDDによる手法は，比較的交差の多い密集したネットに対して有効であると考えられる．

表2:交差数による順序付けの効果

順序付け考慮なし		交差数による順序付け	
BDD節点数	処理時間(sec)	BDD節点数	処理時間(sec)
161468	109.64	95540	53.77
112649	72.67	126841	52.24
85484	106.68	42232	56.51
151497	172.33	61078	36.31
80073	56.90	43478	27.72

表3:交差関係による順序付けの効果

順序付け考慮なし		交差関係による順序付け	
BDD節点数	処理時間(sec)	BDD節点数	処理時間(sec)
161468	109.64	107989	71.08
112649	72.67	81176	48.18
85484	106.68	26793	34.05
151497	172.33	35064	26.67
80073	56.90	22215	20.32

表4:二つの基準と効果

w1/w2	節点数	処理時間(sec)
0	371	0.65
0.5	1058	0.72
1.0	1058	0.72
2.0	1387	0.81
3.0	1388	0.51
3.5	1703	0.39
10.0	1712	0.76
∞	1734	0.42

表5:位置を考慮した順序付けの効果

順序付け考慮なし		位置による順序付け	
BDD節点数	処理時間(sec)	BDD節点数	処理時間(sec)
161468	109.64	31468	21.07
112649	72.67	33773	32.99
85484	106.68	13763	21.75
151497	172.33	21356	20.14
80073	56.90	18587	19.94

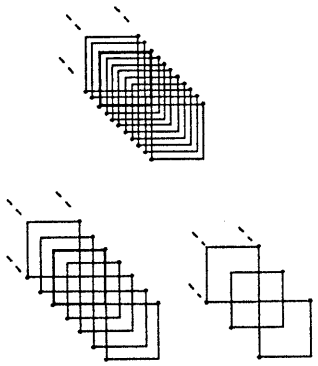

図6　様々な交差数のネット

表6:ネットの交差数とBDD節点数

交差数	BDD節点数	処理時間(sec)
2	228	0.86
10	660	1.16
20	728	1.21
30	620	1.13
36	296	1.04
40	80	0.85

7　おわりに

本稿では，BDDに基づくマンハッタンワイヤリングの最大ネット集合を求める方法について述べた．この方法を計算機上に実装し，実際の問題例についていくつかの実験を行った結果，この手法は，比較的混雑したネットに対して有効であることと、本稿の問題に対して有効なBDDの変数順序付け法は，配線の位置をもとにした順序付け法であることが分かった．

現段階では，ここで示した手法の他手法に対する明確な利点はほとんど明らかになっていない．評価を明確にするためには，分枝限定法等の従来手法との比較を行うべきであろう．

＜謝辞＞

計算機実験に関してご協力いただいたNTTの湊真一氏にここで深く感謝いたします．本研究は文部省科学研究費補助金:奨励研究(A)05855055(平成5年度)「柔軟性の高いLSIレイアウト設計手法に関する研究」の援助のもとに行われたものである．

＜参考文献＞

[1]R.E.Bryant, "GraphBased Algorithms for Boolean Function Manipulation, "IEEE Trans.Comput., Vol.. C-27, No.8, pp.667-691(1986).

[2]柳谷雅之, "組み合わせ最適化問題のBDDによる解法, "情報処理, Vol.34 , No.5 , pp.617-623., (1993).

[3]増田澄男, 木村晋, 柏原敏伸, 藤沢俊男、"マンハッタン配線問題に関して, "信学技報 CAS83-20

[4]H.imai and T.Asano, "Dynamic Segment Intersection Search with Applications, "Proc.25th Annu, IEEE Symp. Founda. of Comp. Sci, pp.393-402, (1984).

[5]Raghavan, "Manhattan and rectilinearwiring, " univercity of Minnesota, Technical Report 81-5 (1981)

[6]S.Minato, N.Ishiura, .and S.Yajima, "Shared Binary Decision Diagram with attributed Edges for efficient Boolean Function Manipulation, "Proc. 27th ACM/IEEE DAC.,pp.52-57.(1990)

[7]K.S.Brace, R.L.Rudell and R.E.Bryant , "Efficient Implementation of a BDD Package, " Proc.27th ACM/IEEE Design Automation Conference. pp41-45(1990)

[8]S.J.Friedman and KJ.Spowit, "Finding Optimal Variable Ordering for Binary Dicision Diagrams, "24th ACM/IEEE DAC, pp.348-355(1987)

[9]N.Ishiura, H.sawada and S.Yajima "Minimization of Binary Dicision Diagrams Based on Exchanges of Variables, "Proc.IEEE Int Conf . on Computer-Aided Design, pp.472-475(1991).

[10]湊真一, "共有二分決定グラフの「幅」に着目した変数の順序付けによるノード数削減法, "第四回回路とシステム軽井沢ワークショップ, pp.271-276(1992).